# A Multicast Based Bandwidth Saving Approach for Wireless Live Streaming System*

Wenbin Jiang, Hai Jin, Xiaofei Liao, Zhi Yuan
*Services Computing Technology and System Lab*
*Cluster and Grid Computing Lab*
*School of Computer Science and Technology*
*Huazhong University of Science and Technology, Wuhan 430074, China*
*{wenbinjiang, hjin, xfliao}@hust.edu.cn*

***Abstract***

*P2P based live streaming applications are growing up rapidly. However, when extending them to WLANs, the bandwidth bottleneck and the high rate of packet loss are usually the major stumbling blocks. To overcome this issue, a new wireless multicast agent mechanism (WiMA) based on the IP multicast scheme for buffer management and scheduling is proposed. First of all, a wireless agent selection method is presented to choose an appropriate agent for WiMA. Then this agent gains media data by interacting with wired neighbor peers as a common P2P node, and transmits these data to other wireless peers in the WLAN by means of multicast pushing and multicast patching. A normal wireless peer requests data according to the strategy of "emergent ones first". Experimental results show that the approach proposed can significantly save bandwidth in WLANs with acceptable start delay and satisfied playing continuity.*

## 1. Introduction

At present, P2P live streaming systems have gained large accomplishment in the delivery of media data and the control of service quality by applying application-level multicast. Its main idea is to realize multicast function by enhancing the functions of end systems without changing the backbone of the Internet, while maintaining the model of “Unicast, try best to deliver”. It has been used widely since it brings few changes to all layers below the application layer of the network. P2P scheme can enhance the extensibility of the large scale network applications effectively and extricate them from the big headache of the bandwidth bottleneck of the servers in client/server mode. Many P2P live streaming systems come forth such as PeerCast [1], CoolStreaming [2], AnySee [3], PPLive [4], PPStream [5], etc. However, there are few systems that consider extending P2P mode to wireless environments.

Nevertheless, a new era of ubiquitous computing is coming with the rapid development of computation, networking and communication technology. Many intelligent mobile devices such as PDA, smart phone, pocket PC become more and more popular. Their abilities of computing and processing also become more and more powerful along with numerous wireless broadband transfer means. All these promote the trend of wireless live streaming [6]. However, the realization of live streaming in wireless network still faces huge challenges such as the low processing ability of mobile devices, the instability of wireless network and the conditionality of the bandwidth, compared with wired network and desktop computers.

* This paper was supported by the National Basic Research Program of China (973 Program) under grant No. 2006CB303000. It also was supported in part by China National Natural Science Foundation (NSFC) Grants No. 60642010, 60703050, the Research Fund for the Doctoral Program of Higher Education Grants No. 20050487040.

Although some mobile TV, cell phone TV systems based on 2.5G/3G networks are more and more popular, most of them are mainly based on exclusive mobile communication networks with high cost and complicated maintaining technology. And most of them are based on client/server mode. Moreover, it is not easy to bind the existing P2P live streaming systems to the above frameworks.

To improve the mobility of the P2P live streaming system, the idea of extending it into wireless LAN (WLAN, WiFi) [7] is reasonable and desirable. Moreover, since WiMAX [8] is a technology deriving from WiFi, it will be relative easy to move P2P live streaming system from WLAN to WiMAX. As we all know, WiMAX is now emerging as a great competitor with higher bandwidth to traditional 3G networks. It will provide more mobility and convenience to P2P live streaming system with high stream quality.

However, it is unreasonable to extend the existing P2P live streaming systems for WLANs without any improvement due to the low bandwidth, the high rate of packet loss of WLAN. If the number of peers accessing the system in the WLAN increases, the application-level multicast realized by peer-to-peer unicast links will consume much bandwidth, which is a big overhead for WLAN.

To overcome this issue, a new wireless multicast agent mechanism (WiMA) is presented here to provide stable and fluent streaming services for wireless peers. Based on AnySee that is a practical P2P live streaming system developed by our lab, a new hybrid living streaming system is realized by applying WiMA. It applies IP multicast to achieve media data delivery in WLAN, which reduces the overhead of the network significantly. Meanwhile, based on the features of media data, the approaches of multicast pushing and patching are designed to improve the reliability of data transfer and assure the continuity of stream playing.

This paper is organized as follows. In Section 2, an overview of related work is provided. Section 3 discusses the design of wireless multicast agent mechanism in detail. Experimental results are shown in Section 4. Finally, Section 5 draws the conclusion and talk about some future work.

## 2. Related Work

For live streams delivery, a lot of significant work has been done. Generally, there are two major representative directions.

On one hand, research in mobile live streaming systems is developing rapidly, and many applications come forth.

MobiTV [9] has realized live streaming services in the cellular networks of American vendors Sprint and Idetic. Till early 2006, the number of MobiTV users has exceeded two millions. Moreover, it is worthwhile to notice that the MobiTV has paid considerable attention to WLAN and WiMAX Radio Access Technologies (RATs) and declared that it will support multicast and WiMAX and apply these technologies to deploy new network. However, the new schedule is still a blueprint. Mobile live streaming system based on WiFi or WiMAX is rare to see.

CMT-GGTV, the mobile streaming platform of GGTV [10] could provide various media services for users using mobile ends via 2.5G mobile networks such as GPRS, CDMA1X and 3G ones such as WCDMA and CDMA2000. Live streaming, Video on Demand (VOD) were its representative services. However, the platform still applied client/server mode.

On the other hand, research about P2P based live streaming technology over wired Internet is also rapidly gaining momentum.

PeerCast was developed by Stanford University. To lessen the impact of the dynamic transformation of the P2P overlay structure to the quality of media streaming, it applied redundant connections. It reduced transmit delay between peers by limiting the number of nodes that each node serves. Presently, PeerCast is mainly used in small scale Internet radio broadcast and has run on Gnutella network successfully.

Coolstreaming/DONet was a typical P2P live streaming system that applied mesh based application layer multicast. The main idea of this approach was making use of light-weight Gossip [11-13] protocol to construct an overlay network in the application layer. However, wireless network were not considered.

The AnySee system was another representative P2P live streaming system that used self-adaptive delivery and management strategy. It achieved high scalability and reliability and has attracted more than 10,000 users, and served more than 2,000 concurrently online users. The next step in AnySee is to take the WLAN into account.

Also, some commercial P2P streaming systems such as PPLive, PPStream, have achieved great business success.

Generally, several application-level schemes for optimizing live streaming overlays have been achieved, including tree-based [14-15], mesh-based [2][16][17], and hybrid overlays [18]. However, systems above mainly considered using application-level multicast to solve the problem of media delivery, which still consumed considerable bandwidth, and make it unsuitable for pervasive environments. If IP multicast were used here, more bandwidth would be saved.

The idea of using IP multicast to solve some resource conflict problems in some special environments has caught some researchers' attention. Yoid [19] and Host Multicast [20] are representative examples. The main idea of them was: in local or small scale networks (called "IP multicast islands") that supported IP multicast, the IP multicast was used to transfer data. To connect these "islands", similar to P2P mode, application-level multicast was applied. However, these approaches have paid little attention on live streaming. Since the live streaming applications in WLAN is still a major issue, a novel IP multicast agent mechanism is desired.

## 3. Design of Wireless Multicast Agent Mechanism

The P2P scheme has effectively solved the problems of the heavy overhead and the bandwidth bottleneck of the server in the traditional client/server mode in the wired networks. However, when extending P2P scheme to WLAN, some new challenges come out, including the limitation of wireless network bandwidth, the high rate of packet loss and the distinct mobility of peers. If the media data were still exchanged among wireless peers by application-level multicast protocols that usually used unicast in the network layer, bandwidth would be used up rapidly. If these peers request an identical channel, bandwidth will be wasted on much redundant data transfers.

To solve the problems mentioned above, this paper proposes a wireless multicast agent mechanism named WiMA, which will support more online users in WLAN. By using IP multicast to achieve media data delivery, the overhead of WLAN network can be reduced significantly. Meanwhile, in accordance with the features of media data, the methods of multicast pushing and multicast patching are proposed here, which can improve the reliability of data transfer obviously and guarantee the continuity of stream playing effectively. Fig.1 is the network topology of this hybrid live streaming system.

Peers in the wired network get media data by application-level multicast from their neighbor peers or the center servers. However, for the wireless peers (wPeers) in the WLAN, an agent will be selected at first. On one hand, it works as a normal peer like peers in the wired network to get media data from the wired network. On the other hand, it serves as a proxy for other wPeers that want the same channel. The others get media data from the agent instead of outside peers.

Of course, a unified IP multicast address must be set for each channel before the multicast agent starts to work if it has not be configured. Moreover, when a new wPeer joins in this channel, it receives multicast media data according to multicast address. If this wPeer has potential to be an agent, a new joining agent selection process will be started. As IP multicast has no acknowledgement mechanism unlike the traditional wired P2P scheme, reliable transfer can not be ensured. So some patching mechanisms are desired.

Following is the description of the WiMA in detail. First of all, the selection method for the appropriate wireless agent should be decided.

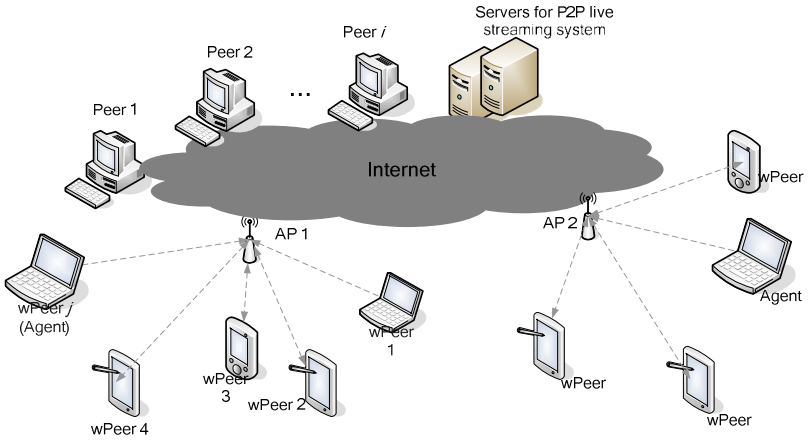



**Fig. 1** System network topology

### 3.1. Multicast Agent Selection

Many factors such as the intensity of the radio signal, the network traffic and the power of the CPU all affect the ability of a wireless peers to be an agent. Especially, the signal intensity is the key fact that decides the rate of packet loss and the stability of the connection to wireless AP (Access point). Here, in the multicast agent selection, an agent competition weight value $W$ is used to estimate the ability of a wPeer that is considered to be an agent. Every candidate wPeers compute their weight values and encapsulate them into the heart-beating packets (HBPs) and multicast them to others periodically. When they receive HBPs from others, they compare all weight values and find the best one. In HBP, there are 5 items that are related to agent selection, including the general universal identifier (GUID), the competition weight value $W$, the best agent ID (BestGUID), the first slot number (FSN), and the agent flag IsAgent. Every wPeer maintains a table of competition weight values called agent competition table (ACT), and updates it when getting a new HBP. Following discusses a wPeer how to compute the competition weight value $W$.

Let $W_s$ denote the contribution of the signal intensity to the weight value. It can be computed by

$$W_s = M_p / M_T \tag{1}$$

Here, $M_T$ is the theoretical number of the HBPs that this wPeer should receive from other wPeers. And the $M_p$ is the actual number of the HBPs received. Obviously, $0 \le W_s \le 1$. Since HBPs are multicasted periodically with the period $T_b$. $M_T$ can be obtained by

$$M_T = N \cdot T_{\mathrm{d}} / T_{\mathrm{b}}$$

Where, $T_d$ is total time of the current wPeer being alive. $N$ is the number of other wPeers that are talking to this wPeer.

Moreover, the $W_l$ is used here to denote the stability of the wPeer. It can be estimated by

$$W_l = T_d/(T_d+a) \tag{2}$$

From above, $0 \le W_l \le 1$. The longer the $T_d$ is, the more the $W_l$ is close to 1, which means the wPeer is more stable. Here, $a$ is an adjustable factor that affects the converging speed of $W_l$.

To make the wPeer that plays ahead has higher priority, $W_{fsn}$ is used to denote its affection.

$$W_{fsn} = \begin{cases} \sum_N (FSN_0 - FSN_i) \Big/ \sum_N |FSN_0 - FSN_i|, & \exists i \quad FSN_0 \neq FSN_i \\ 0, \quad Otherwise \end{cases} \tag{4}$$

Here, $FSN_0$ is the first slot number of this wPeer. $FSN_i$ is the ones of other wPeers. From the above equation, we can get $-1 \le W_{fsn} \le 1$. Finally, the total normalized weight value can be calculated by

$$\hat{W} = \alpha \cdot (W_{fsn} + 1)/2 + \beta \cdot W_s + (1 - \alpha - \beta) W_l \tag{5}$$

Where, $\alpha, \beta$ ($0 \le \alpha, \beta \le 1$) are the adjustable factors. Since the weight value is coded in one byte, the final weight value is scaled by

$$W = 256\hat{W} \tag{6}$$

Fig. 2 gives the flow of the new agent selection process. It is triggered by the coming HBP from other neighbors. When one wPeer receives a HBP from other wPeer, it firstly calculates its own weight value by equation (6) and updates its HBP and ACT. Then judge whether this HBP is from an agent or not. If yes, it means that the agent has already existed or is new one selected right now. If the agent is a new one selected, the current wPeer will set the BestGUID as the new agent and request media data from it. Otherwise, it will go ahead without change. If the received HBP is not from an agent, and the wPeer has not received HBP from an agent for a considerable long time, it means that the agent maybe has been disabled. Then the wPeer should consider voting a new one. Of course, the wPeer will wait for an appropriate time span to make sure the old agent failed assuredly before it starts to vote. This time span is usually about 4~5 heart beating periods practically. If the old agent has been verified to be disabled, this wPeer compares its weight value with ones of all others in the ACT and finds the BestGUID. If it own is the BestGUID and makes sure that it has been accepted by more than 2/3 wPeers according to the historical HBP records, it sets itself as the new agent and updates HBP, and tells others by HBP multicast. If not, it goes to next cycle. By above approach, the new agent can be obtained efficiently. And the system recovers to the stable status rapidly.

Here, a multicast agent module is used for the buffer management of multicast. Fig.3 depicts the function components of this module. They are multicast pushing, multicast patching, and heart beating (HB) management. Multicast pushing and multicast patching

components are in charge of transferring media data to other wPeers. The heart beating mainly assists the new joining wPeer to discover the agent and initialize its buffer.

In the following, main attention is paid on multicast pushing and multicast patching.

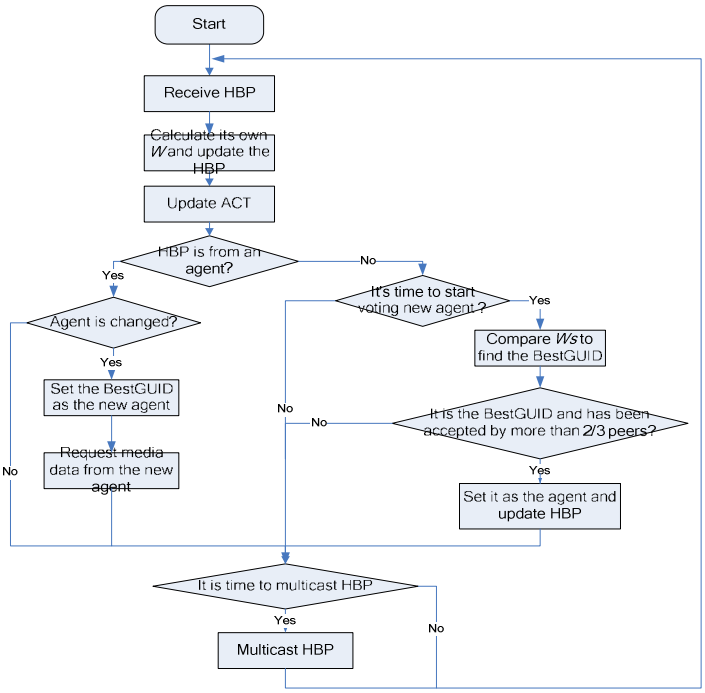



**Fig. 2** Multicast agent selection

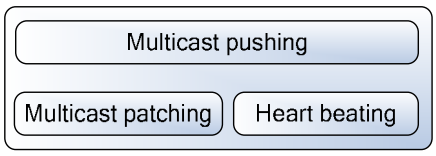



**Fig. 3** Module of Multicast buffer management

### 3.2. Data multicast combining pushing and patching

Here, on the one hand, most media data of one wPeer are received by multicast pushing when it is in stable playing status. On the other hand, multicast patching is used as supplement during the procedure of buffer initialization of the wPeer, and in the stable playing status when some media data slots are lost since the IP multicast has no acknowledgement mechanism itself. Here the buffer is formed by a series of data slots that are the basic units of scheduling. The procedure of data buffering by multicast pushing and multicast patching is shown in Fig.4.

When a new wPeer joins the current channel of live streaming, it will receive some media data from the agent by multicast pushing soon after the buffer initialization. However, these data mainly are located at the tail of the agent's buffer, which is also the end of the wPeer's buffer. For the empty foreside of the buffer of the wPeer, it is requested from the agent by

multicast patching. When the normal wPeer has enough media data in its buffers, it begins to play. While the wPeer continues to request multicast patching data till the foreside of the wPeer buffer is full. Then, the buffering procedure becomes stable, and the buffer rolls ahead with the program playing. The multicast pushing becomes the main way to provide the subsequent media slot packets desired to fill the empty tail part after the buffer rolling. The multicast patching is mainly used to request the lost data slots caused by network traffic or some other reasons once more. Following is the detailed buffer schedule schemes of the normal wPeer and the agent.

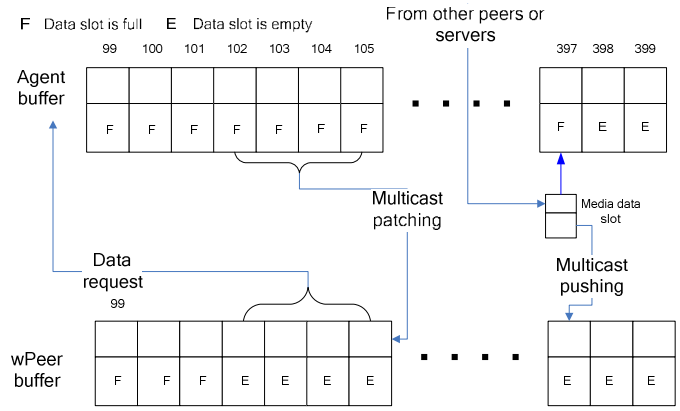
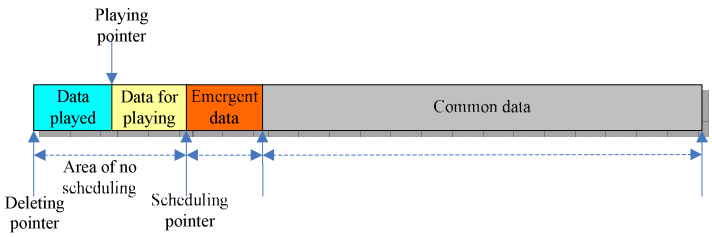


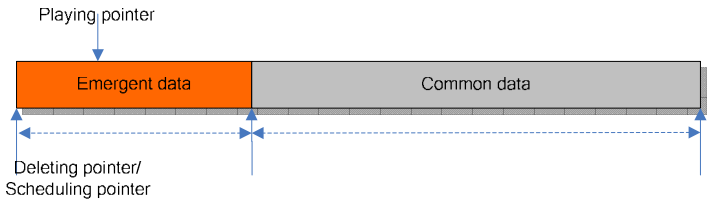

**Fig. 4** Data multicast combining pushing and patching

### 3.3. Schedule scheme of normal wPeer

First, the segmentation of the buffer should be decided. Here the buffer of a normal wPeer is divided into several segments according to the emergency level of the corresponding data. The more emergent the data slot is, the earlier the wPeer requests it from the agent. Of course, for the emergent data slots that are to be deleted on the instant in the agent buffer, they will be transferred by multicast patching first. Generally, the agent tries to multicast the media data it has to the whole wireless network and makes the buffers of wPeers close to the one of the agent enough. Fig. 5 (a) shows the inner structure of the normal wPeer's buffer.




(a) Normal wPeer



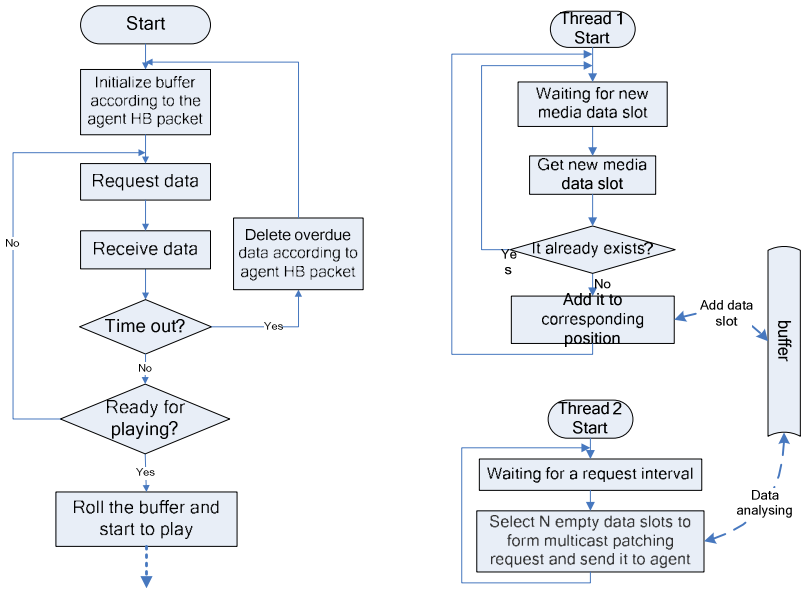

(b) Agent

**Fig. 5** Buffer structure of wPeers and agent

Between the deleting pointer and the playing pointer are some played data slots. These old data are useful when this wPeer is selected as an agent and it will push them to other wPeers. Between the playing pointer and the scheduling pointer are the data ready to be played. The length of this segment is a buffer maintaining period (BMP). This segment will not be requested anymore since the wPeer must hand up it to the player at once to keep the playing continuous. An emergent data segment is following the scheduling pointer and its length is also a BMP. It is the most emergent with highest priority since it will be sent to player in the next BMP.

Fig. 6 shows the workflow of the normal wPeer based on the above buffer segmentation. The scheduling algorithm has two stages. The first is the buffer initialization. In this stage, the wPeer receives the HB packet from the agent that multicasts it periodically. The wPeer produces multicast patching requests according to the HB packet and the status of its buffer and sends it to the agent. Then the wPeer starts to receive data slots from agent for some time. If the buffer is ready for playing in an endurable interval, the wPeer will start to play and roll the buffer. Otherwise, if time out, the wPeer should delete some overdue data slots of the buffer according to the last HB packet to make sure the buffer is consistent with the one of the agent and restart the initialization.

After the initialization, the wPeer goes to the second stage, stable playing stage. In this stage, on the one hand, the wPeer receives new data slots multicasted by the agent and adds them to the corresponding positions of its buffer if they are not included in the buffer yet. On the other hand, after a request interval, the wPeer will select N empty data slots to form a multicast patching request and send it to the agent. The agent will multicast these data slots again according to its scheduling policy discussed below.




(a) Initialization (b) Stable playing

**Fig. 6** Workflow of the normal wPeer

### 3.4. Schedule scheme of agent

The agent also needs to consider the segmentation of its buffer when responding to various requests from normal wPeers. The segmentation of the agent buffer differs a little from the one of the normal wPeer. The emergent data segment begins from the position of the deleting pointer (see Fig. 5 (b)). In other words, the played data segment and the data segment for playing in the normal wPeer buffer both belong to emergent data segment in the buffer of the agent. This ensures that the agent can multicast media data to others as much as possible before the data is deleted from the buffer. The buffer of the wPeer can keep up with the one of the agent as consistently as possible.

In order to discuss the schedule of the agent more clearly, some concepts should be explained in advance.

- ***First slot number*** **(FSN):** it is the serial number of the first data slot in the buffer of the wPeer or the agent.
- **The *deadline of a data slot requested* (DDSR)**: it means that the corresponding data slot is useful for a normal wPeer if it is received before the deadline. When a normal wPeer sends a multicast patching request to the agent, the request includes its FSN and the *requested data slot number* (RDSN), the DDSR of this data slot can be calculated by subtracting FSN from the RDSN.
- **The *number of requests for one data slot* (NRDS)**: since the agent may receive multicast patching requests for the same data slot from different wPeers. The agent will count them up by NRDS, which is useful for agent to decide how to respond the requests from wPeers.

Fig. 7 depicts the work flow of the agent. When the agent receives a multicast patching request from a wPeer, it updates the DDSR and NRDS of the corresponding data slot in the queue of data slots requested (QDSR) with the information from the request. If this data slot has been requested by another wPeer, compare the two DDSRs of the two requests and record

the smaller one. Then the agent sorts the queue according to their DDSRs in ascending order. For those requested data slots whose DDSRs are smaller than the emergent schedule interval (ESI), they will be scheduled at once. For those whose DDSRs are larger than ESI, sort them by their NRDSs in ascending order. The lager the NRDS is, the earlier the corresponding data slot will be scheduled. By the above approach, the most emergent and valuable data can be scheduled first and the bandwidth can be saved effectively.

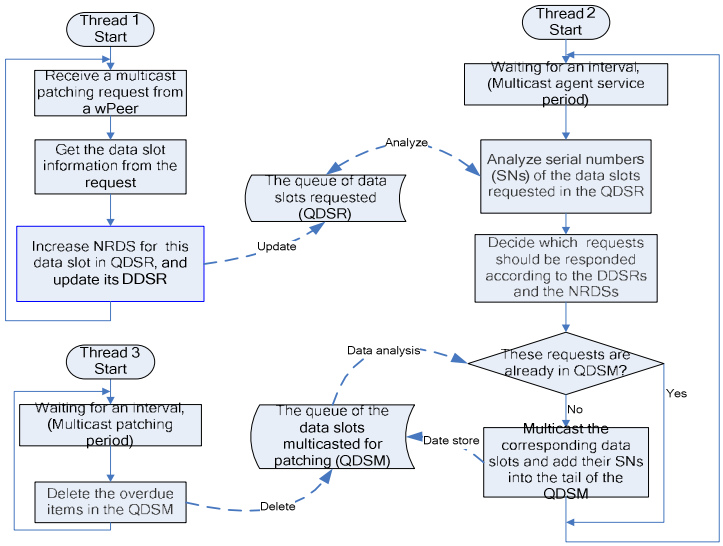



**Fig. 7** Workflow of the agent

Meanwhile, to prevent the agent from over-responding that will aggravate the overheads of wireless network and the agent, the queue of the data slots multicasted for patching (QDSM) is set in the agent here. It records some data slots that have been multicasted just now. When the agent wants to multicast a data slot, it will check the QDSM to know if this slot is in it. If so, it means this slot has been multicasted just now and it's unnecessary to do it again. This QDSM is maintained by a sub thread that deletes the overdue items in QDSM periodically.

Concerning the agent heart-beating mechanism mentioned above. On the one hand, it lets the agent multicast its heart-beating packet that contains its FSN and the snapshot of its buffering to others periodically. It can help newly added wPeers to initialize their buffers faster. On the other hand, when the agent leaves suddenly, it can promote the competition procedure of wPeers for selecting new multicast agent.

## 4. Experiment

To verify the efficiency of the mechanism presented in this paper, some experiments have been done. Experimental hardware equipments consist of 2 mobile PDAs, 3 wireless laptops, 7 PCs (among them 6 are wired including 1 tracker server (TS), 1 broadcast server (BS), 4 wired peers and 1 wireless PC), and 1 wireless AP that links all wireless peers including PDAs, laptops, and wireless PC. It works on 802.11b with 11Mbps bandwidth. More details are listed in table 1. According to the principle of the IP multicast configuration, the address 224.0.0.1 is applied as the IP multicast address. It is customarily used for all hosts that join in the IP multicast group on a subnet.

**Table 1.** Hardware equipments for experiment

| Machine | CPU | Memory | OS | Network link |
|---|---|---|---|---|
| TS (1) | PIV 2.0G | 512M | Win2003 | 100M |
| BS (1) | PIV 2.0G | 512M | Win2003 | 100M |
| Wired clients (4) | Celeron 2.0G | 256M | WinXP | 100M |
| Wireless PC (1) | Celeron 2.0G | 256M | WinXP | 11Mbps |
| AP (WR541G, 1) | N/A | N/A | N/A | 11Mbps |
| laptops (3) | PIV 1.73G | 512M | WinXP | 11Mbps |
| PDAs (2) | SC32442 300M | 128M/64M | WM 5.0 | 11Mbps |

Three typical stream samples (Show in Fig. 8) are used here. Table 2 gives some details of the three. Some comparisons are discussed below.

(1) Kongfu (2) Joke (3 ) Firsttime

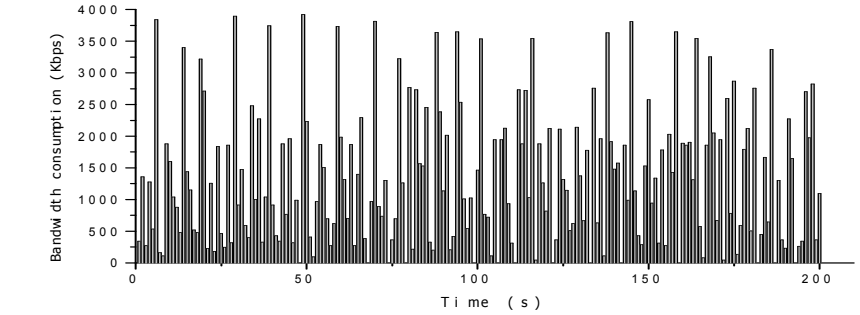
**Fig.8** Three streams for experiment

**Table 2.** Streams for experiment

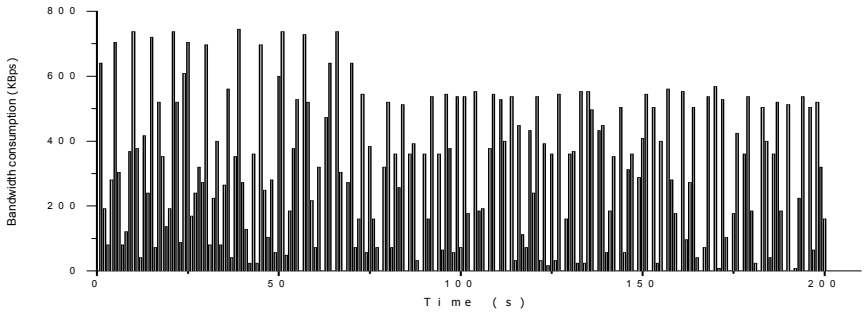
| Name | Resolution | Rate (Kbps) | Length |
|---|---|---|---|
| Kongfu | 320x240 | 240 | 9:35 |
| Joke | 288x224 | 100 | 17:09 |
| Firsttime | 240x176 | 90 | 5:29 |

### 4.1. Bandwidth consumption

The rate of bandwidth consumed will affect the stability and the extensibility of the live streaming system. Fig. 9(a) shows the bandwidth consumption by means of non-multicast for "Kongfu" stream. 5 wireless peers require media data from this channel by normal P2P mode. The peak value of bandwidth consumption in WLAN is close to 4000Kbps, and the average is 1300Kbps. As mentioned above, 802.11b is used here, its theoretical bandwidth is 11Mbps, and its practical bandwidth is about 5Mbps. In other words, if the bandwidth consumption is close or up to 5Mbps, the efficiency of the WLAN will be reduced obviously and the rate of packet loss becomes high. In fact, when one more wPeer join in the same channel, the QoS of this channel decreases immediately. Namely, this mode just can support 5 wPeers.




(a) Non-multicast




(b) Multicast

**Fig.9** Bandwidth consumption of "Kongfu" stream

Fig.9 (b) shows the bandwidth consumption of the same stream by means of WiMA. The average value is only about 280Kbps and the peak value is about 720Kbps. By comparing the two modes, we can see that the bandwidth consumption by WiMA is much lower than that by non-multicast. This mode can support much more wPeers than unicast mode. Practically, when one more wPeers is added into the same channel, the increment of bandwidth consumption is inconspicuous. The conclusions for the two other streams are similar. Bandwidth consumptions of them are given in Table 3.

**Table 3**. Bandwidth consumption comparisons

| Name | Unicast (kpbs) | Multicast (kpbs) |
|---|---|---|
| Kongfu | 1300 | 280 |
| Joke | 604 | 125 |
| Firsttime | 497 | 103 |

### 4.2. Start delay

Start delay is one of the important parameters to measure the QoS of the system. Fig. 10 gives some start delays statistics with different modes. The square denotes the start delays of some laptops by non-multicast. The round denotes the ones of some laptops by multicast. The triangle denotes the ones of some PDAs by multicast. Obviously, the first ones are fastest. The average is about 17s. The second ones are a little longer than the first with the average about 22s. For the third ones, since the network links from PDAs to AP are more unstable and the powers of them are weaker, the start delays (the average is about 28s) are slowest.

Nevertheless, these start delays are acceptable, compared with many other live streaming systems [21].

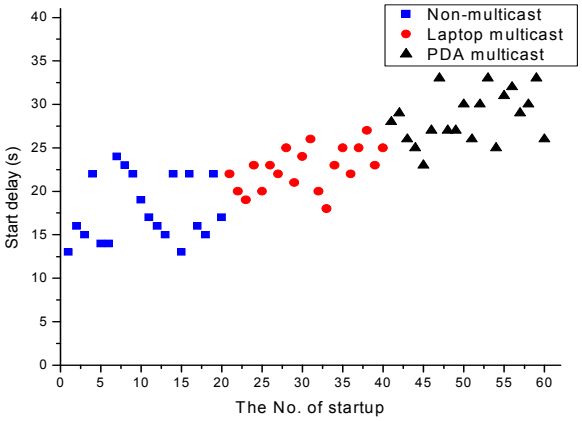
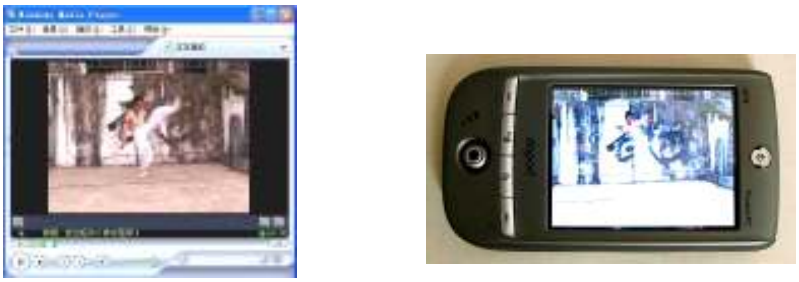



**Fig. 10** Start delays

### 4.3. Buffer fill rate

The higher buffer fill rate means the peer has more effective data for playing. Experiments show that, buffer fill rate by non-multicast is about 96% on average. However, under the mechanism of multicast agent, buffer fill rate is about 92% on average. Although it is a little lower than the former because of the packet loss, the limited ability of patching, it still can ensure the stability of the peer adequately, which has been verified by the experiments. The table 4 shows the details of some peers' buffer fill rates for the three streams. Additionally, the PDAs have lower buffer fill rates than laptops because the special features of the PDAs mentioned above.

Fig. 11 gives the playing interfaces of different wireless devices of the stream "Kongfu" in the hybrid live streaming system presented here. (a) is one laptop. (b) is the one of the two PDAs.



(a) Playing in laptop (b) Playing in PDA

**Fig. 11** Playing stream "kongfu" in mobile devices by WiMA

**Table 4.** Buffer fill rate by WiMA

| Normal wPeers | | Kongfu | Firsttime | Joke |
|---|---|---|---|---|
| PDA | 1 | 89% | 90% | 91% |
| | 2 | 87% | 88% | 92% |
| Laptop | 3 | 93% | 94% | 96% |
| | 4 | 91% | 95% | 98% |
| | 5 | 92% | 95% | 94% |

Generally, the experiments have shown that the stream data can be obtained by wireless peers effectively and played smoothly on various mobile devices by WiMA, and the traffic of the wireless network is affected little by the number of users.

## 5. Conclusion

This paper tackles the difficulties caused by extending P2P live streaming system to WLAN by proposing a wireless multicast agent mechanism WiMA for buffering and scheduling. It points out that in multicast agent mechanism, the problems focus on how much and which data should be scheduled. Some algorithms and strategies about multicast buffering schedule are expounded from two aspects: normal wireless peers and multicast agent. The experimental results show that the method presented in this paper can solve the problem of bandwidth bottleneck in wireless network effectively, and increase the number of users in WLAN significantly. One of the further work focuses on tackling the issue of the streaming service maintaining mechanism when one wireless peer moves from one AP domain to another. In the WLAN protocol, there is no QoS mechanism to guarantee the streaming continuity for cross-domain movement. Some buffering and scheduling schemes in the moving peers and the P2P overlay should be studied, which assures the moving peers have enough data to keep their streams playing continuously when they leave some WLAN AP, and can get back to the P2P overlay as soon as possible when they obtain another one.

## 6. References

[1] H. Deshpande, M. Bawa, and H. Garcia-Molina, “Streaming Live Media over Peers”, Technical Report, Stanford University, 2001. pp.105-122.

[2] X. Zhang, J. Liu, B. Li, and T.-S. P. Yum, “CoolStreaming/DONet: A Data-driven Overlay Network for Peer-to-Peer Live Media Streaming”, Proceedings of IEEE INFOCOM'05, Miami, FL, USA, 2005, pp.2102-2111.

[3] X. Liao, H. Jin, Y. Liu, and L. M. Ni, “Scalable Live Streaming Service Based on Inter-Overlay Optimization”, IEEE Transactions on Parallel and Distributed Systems, Vol.18, No.12, December 2007, pp.1663-1674

[4] PPLive [Online]. Available: http://www.pplive.com

[5] PPStream [Online]. Available: http://www.ppstream.com

[6] J. Vass, S. Zhuan, and X. Zhuang: “Scalable, Error-Resilient, and High-Performance Video Communications in Mobile Wireless Environments”, IEEE Trans. Circuits and Systems for Video Technology, July 2001, pp.833-847.

[7] G. Gaertner and V. Cahill, “Understanding Link Quality in 802.11 Mobile Ad Hoc Networks”, IEEE Internet Computing, Vol.8, No.1, January 2004, pp.55-60.

[8] C. Hoymann, “Analysis and performance evaluation of the OFDM-based metropolitan area network IEEE 802.16”, Computer Networks, Vol.49, No.3, 19 October 2005, pp.341-363.

[9] H. Knoche and J. D. McCarthy. “Designing mobile interaction: Design requirements for mobile TV”, Proceedings of MobileHCI’05, ACM Press, September 2005, pp.69-76.

[10] GGTV Website. Available: http://www.ggtv.com.cn
[11] S. Banerjee, S. Lee, B. Bhattacharjee, and A. Srinivasan, "Resilient multicast using overlays", ACM SIGMETRICS Performance Evaluation Review, 2003, 31(1), pp.55-62
[12] P. Eugster, R. Guerraoui, A.-M. Kermarrec, and L. Massoulie, "From epidemics to distributed computing", IEEE Computer, 2004, 37(5), pp.60-67
[13] A. J. Ganesh, A. M. Kermarrec, and L. Massoulie, "Peer-to-peer membership management for gossip-based protocols", IEEE Trans. Computers, Vol.52(2), pp.139 - 149
[14] V. N. Padmanabhan, H. J. Wang, P. A. Chou, and K. Sripanidkulchai, "Distributing Streaming Media Content Using Cooperative Networking", Proceedings of the International Workshop on Network and Operating System Support for Digital Audio and Video, 2002, pp. 177-186
[15] D. Pendarakis, S. Shi, D. Verma, and M. Waldvogel, "ALMI: An Application Level Multicast Infrastructure", Proceedings of Usenix Symposium on Internet Technologies and Systems, 2001
[16] M. Hefeeda, A. Habib, B. Botev, D. Xu, and B. Bhargava, "PROMISE: Peer-to-Peer Media Streaming Using Collectcast", Proceedings of the 11th ACM International Conference on Multimedia, MM'03, 2003, pp. 45-54
[17] X. Jiang, Y. Dong, and X. D. B. Bhargava, "GNUSTREAM: A P2P Media Streaming System Prototype", Proceedings of IEEE ICME, 2003, Vol. 2, 6-9 July 2003 pp. 325-8
[18] Z. Zhang, S. Shi, and J. Zhu, "SOMO: Self-Organized Metadata Overlay for Resource Management in P2P DHT", Proceedings of IEEE IPTPS, 2003, pp.170-182
[19] P. Francis, "Yoid: Extending the Multicast Internet Architecture", White paper, http://www.aciri.org/yoid/, April 2, 2000.
[20] B. Zhang, S. Jamin, and L. Zhang, "Host multicast: a framework for delivering multicast to end users", Proceeding of IEEE INFOCOM, 2002. pp.1366 - 1375
[21] X. Hei, C. Liang, J. Liang, Y. Liu, K.W. Ross, "A Measurement Study of a Large-Scale P2P IPTV System", Multimedia, IEEE Transactions on, 2007, Vol. 9 (8), pp.1672 – 1687

## Authors

**Wen-Bin Jiang**

He received a Ph.D. degree in communication and information system from Huazhong University of Science and Technology (HUST), China, in 2004. Now, he is an associate professor of the School of Computer Science and Technology at HUST. His research interests focus on ubiquitous computing, computing system virtualization, P2P computing, etc. He is a member of the IEEE and the IEEE Computer Society.

**Hai Jin**

He received a Ph.D. degree in computer engineering from Huazhong University of Science and Technology (HUST), China, in 1994. Now He is a professor of the School of Computer Science and Technology (SCST) at HUST. He is also the Dean of the SCST of HUST. He is the chief scientist of the largest grid computing project, ChinaGrid, in China and the chief scientist of national 973 basic research project in China. His research interests include grid computing, virtualization, peer-to-peer computing, etc. He is a senior member of IEEE and the IEEE Computer Society and member of ACM.

**Xiao-Fei Liao**

He received a Ph.D. degree in computer science and engineering from Huazhong University of Science and Technology (HUST), China, in 2005. He is now an associate professor in the School of Computer Science and Technology of HUST. His research interests are in the areas of virtualization, peer-to-peer system, cluster computing and streaming services. He is a member of the IEEE and the IEEE Computer Society.

**Zhi Yuan**

He is now a Ph.D. candidate in the School of Computer Science and Technology of Huazhong University of Science and Technology, China. His research interests mainly focus on p2p computing, mobile streaming, et